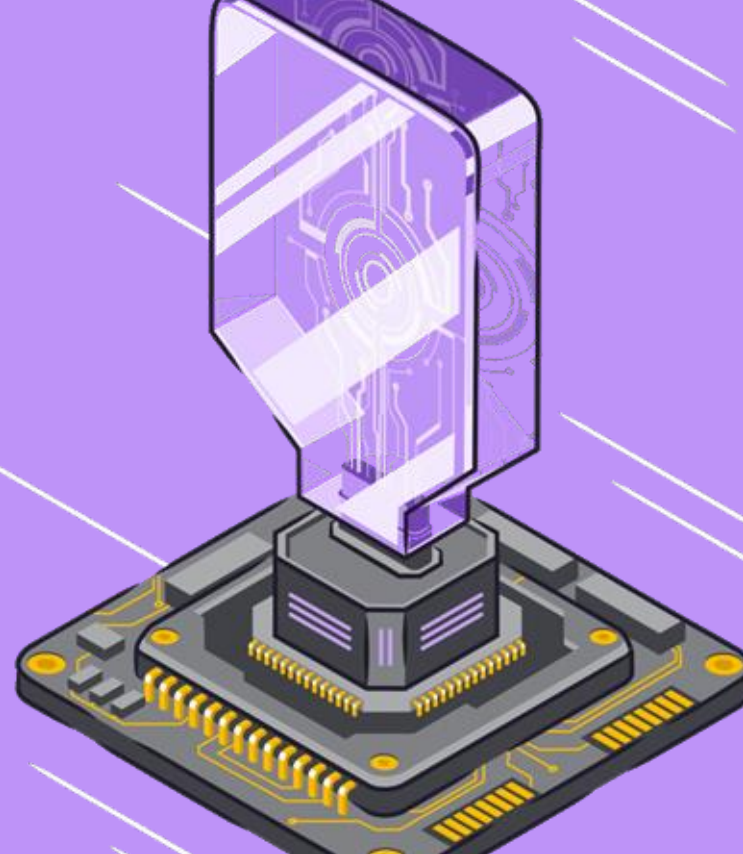




DevOps

DGTL



Как превратить хаос в контекст, развивая Harness

Харченко Евгений, руководитель отдела



Всем привет!
Я — Женя Харченко,
руководитель отдела, в душе инженер



7+ лет
в Райффайзен Банке

Начинал с инженера техподдержки ServiceDesk,
потом работал как Engineer и Leading Engineer



Сообщество

Построил внутреннее профессиональное
DevOps-сообщество с 1500 участников



Мои достижения

Член программного комитета DevOpsConf



Роль в банке сейчас

Руководитель отдела по развитию практик в разработке
и эксплуатации, в который входит 5 команд

План выступления



01 Почему мы говорим про AI и контекст

02 Где в организации теряется знание

03 Как MSH собирает и связывает технологический ландшафт

04 Как это приводит к Digital Twin и AI Harness





DevOps

DGTL

Погружение В КОНТЕКСТ



Инженеры всегда были хранителями знаний



Документация

Мы писали документацию в GitLab и Confluence, вели runbooks — фиксировали, что делать и как устроен ландшафт вокруг наших систем



Контекст

Мы знали процессы вокруг систем и держали в голове их взаимосвязи



Эксплуатация

Если что-то ломалось — разбираться приходилось DevOps'ам. Чаще всего больше никто не знал, как это чинить

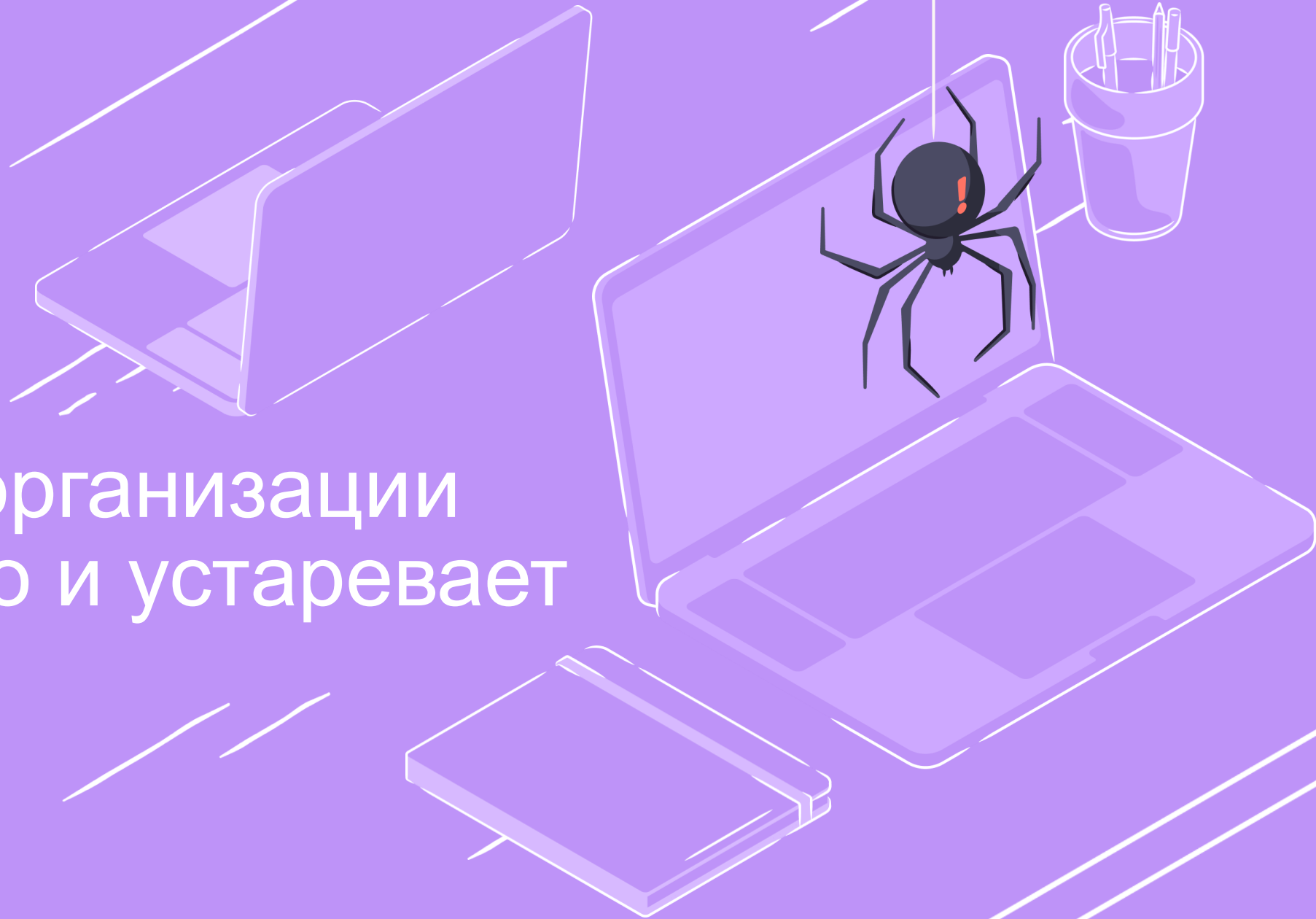




DevOps

DGTL

Знание об организации
разрозненно и устаревает



Данные о ландшафте разрознены



GitLab

Репозитории, CI/CD, исходный код и meta-info.yaml



Порталы и Wiki

Confluence, Insider, Backstage, API Portal и локальная документация



Чаты

Ответы и договоренности часто остаются в тредах



Service Management

SimpleOne, CMDB, справочники и данные об эксплуатации



Что это стоит команде



Документация устаревает

Реальность в коде уходит вперед быстрее, чем правят Wiki



Связи неочевидны

Кто кого вызывает и что сломается при изменении — держится на памяти людей



Владельцев ищут вручную

Чтобы задать один вопрос по API, нужно найти команду и человека



Риски растут

Интеграции замедляются, архитектурные решения принимаются на неполной картине



С приходом AI всё поменялось



Контекст нужен не только людям

Теперь им должны пользоваться ассистенты, агенты и автоматизированные сценарии

AI без контекста слеп

LLM может красиво ответить, но без данных о реальном ландшафте не понимает организацию



Нужен доверенный слой

Актуальные системы, API, владельцы, зависимости, стандарты и ограничения

MSH как основа Harness

Платформа это источник корпоративного контекста для AI-агентов

Harness



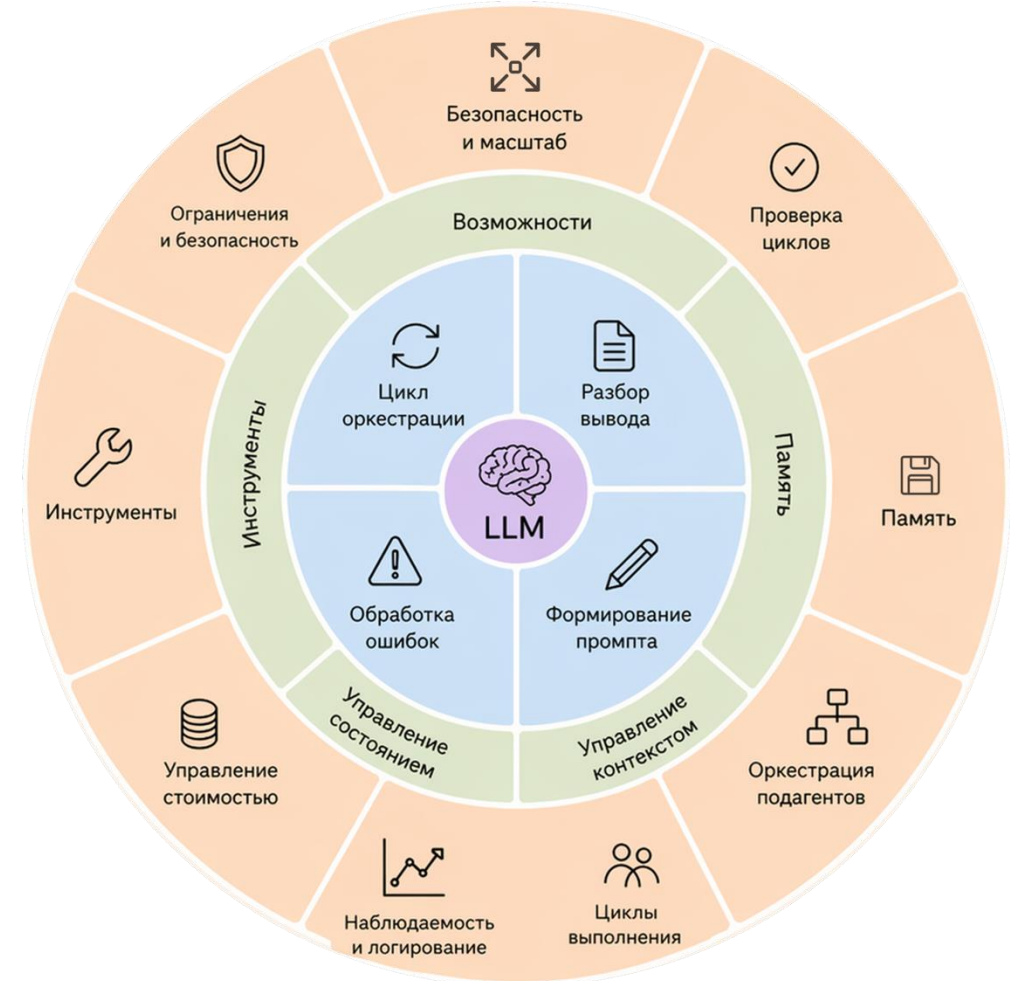
Если модель — это процессор,
то харнесс — операционная
система вокруг него

Харнесс объединяет все,
что окружает LLM-модель

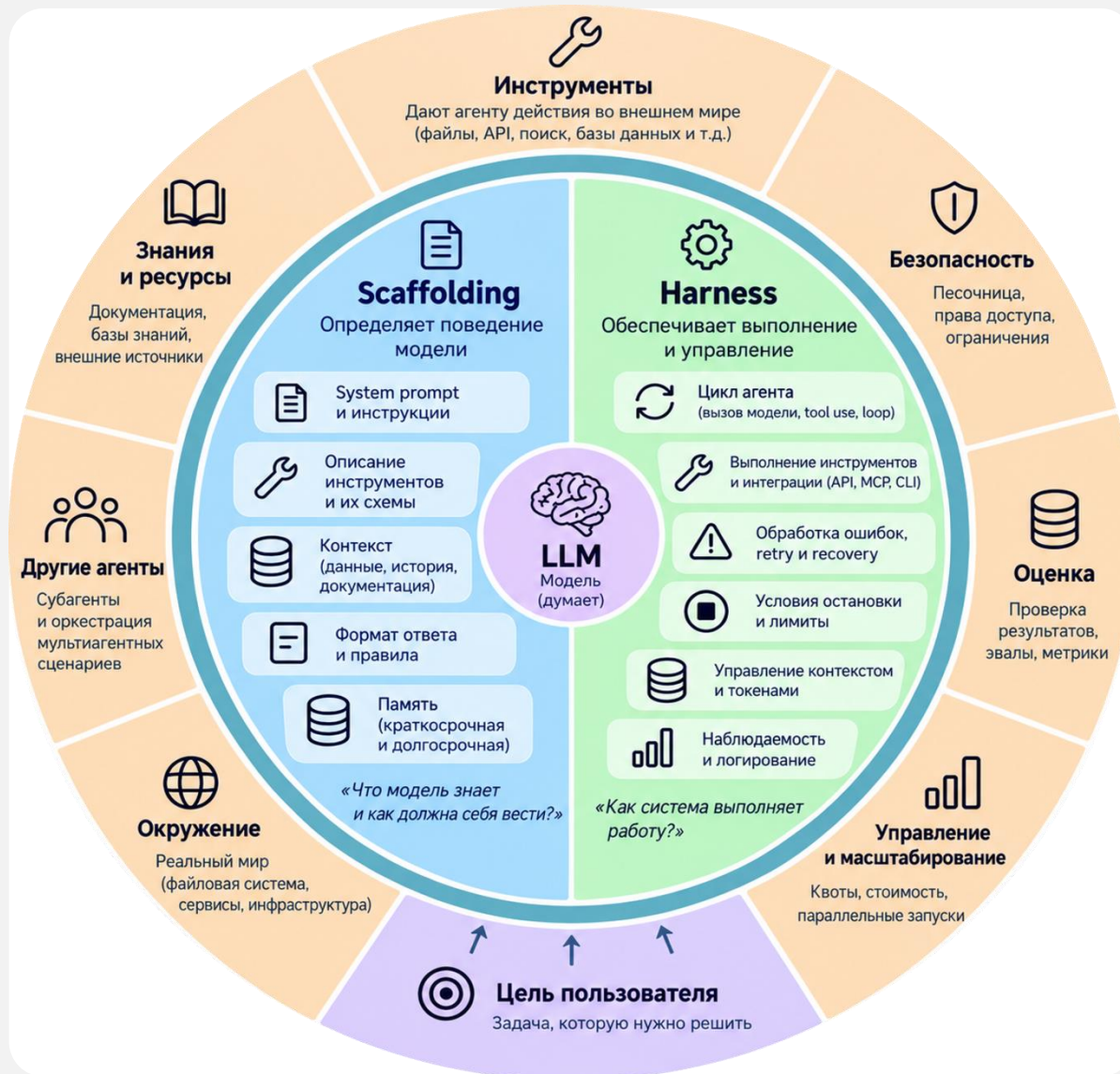


Mitchell Hashimoto,
сооснователь HashiCorp

[Читать статью](#)



Harness | Scaffolding



Harness

Исполнительный слой, который запускает цикл агента, выполняет инструменты, управляет ошибками, ограничениями, контекстом и всей логикой выполнения.

Это то, что делает агента работающим в реальности.

Scaffolding

Поведенческий слой вокруг модели. Формирует её восприятие, знания, правила и доступные возможности.

Это контекст и инструкции модели.

Scaffolding говорит модели, как думать и что ей доступно.

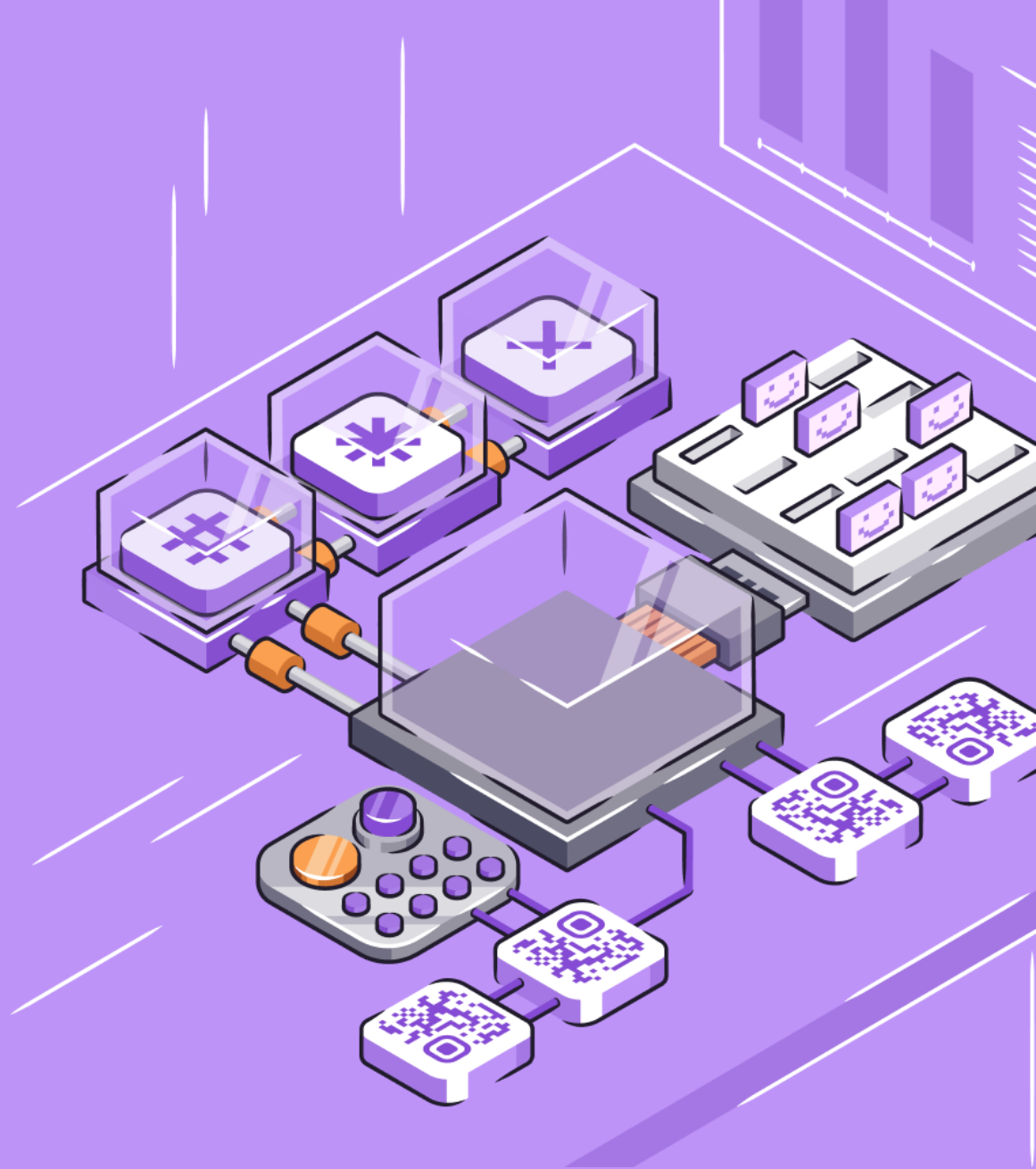
Harness заставляет весь механизм работать в реальности.



DevOps

DGTL

Middleware Service Hub слой контекста системной архитектуры



Почему обычного каталога уже недостаточно



До бума AI у нас уже был Backstage и использовался в основном как API-портал

Каталог компонентов полезен, но закрывает только часть задачи.

Нужна платформа, которая умеет:

- 1 – Собирать
- 2 – Валидировать
- 3 – Связывать
- 4 – Объяснять

The screenshot shows the Backstage Service Catalog interface. At the top, there's a green header with the text "Good afternoon, Guest!" and "Backstage Service Catalog". Below the header, there are tabs for "SERVICES", "WEBSITES", "LIBRARIES", "DOCUMENTATION", and "OTHER". The "SERVICES" tab is selected. On the left side, there's a sidebar with a search icon, a home icon, a puzzle piece icon, a list icon, a cube icon, a target icon, a dollar sign icon, and a bell icon. The main content area is titled "Services" and has a "SUPPORT" link. It shows a list of "Owned (7)" services. The services are listed in a table with columns: NAME, SYSTEM, OWNER, LIFECYCLE, DESCRIPTION, TAGS, and ACTIONS. The services are: artist-lookup, playback-order, searcher, shuffle-api, wayback-archive, wayback-archive-storage, and wayback-search. The tags for each service are: artist-lookup (java, data), playback-order (java, data, playback), searcher (go), shuffle-api (go), wayback-archive, wayback-archive-storage, and wayback-search. The actions for each service are: edit, delete, and star.

NAME	SYSTEM	OWNER	LIFECYCLE	DESCRIPTION	TAGS	ACTIONS
artist-lookup	artist-engagement-portal	team-a	experimental	Artist Lookup	java, data	[edit] [delete] [star]
playback-order	audio-playback	user:guest	production	Playback Order	java, data, playback	[edit] [delete] [star]
searcher		user:guest	production	Searcher	go	[edit] [delete] [star]
shuffle-api	audio-playback	user:guest	production	Shuffle API	go	[edit] [delete] [star]
wayback-archive		team-a	production	Archive of the...		[edit] [delete] [star]
wayback-archive-storage		team-a	production	Storage...		[edit] [delete] [star]
wayback-search		team-a	production	Search of the...		[edit] [delete] [star]

Что такое Middleware Service Hub (MSH) ?



MSH

☰

DevOps Community

Главная

Поиск

Аналитика

Системы

Компоненты

API

Источники

Ресурсы

Команды

Группы поддержки

Домены

Технологии и инновации > devPlatform > MSH Portal

Q Что угодно

⌘ + K

Редактор meta-info.yaml

← MSH Portal

Поиск и просмотр компонентов программного обеспечения и связей между ними

MSH Portal

Middleware Service Hub Portal, представляет из себя каталог компонентов информационных систем нашего банка. Так же он строит связи между различными сущностями, такими как:

- Домены
- Команды
- Информационные системы
- Компоненты
- API
- Ресурсы
- Проекты Gitlab

На данном портале можно найти интересующую сущность (например, API) и посмотреть, к какой системе она относится, какая команда за неё отвечает и в каком проекте GitLab ведётся её разработка. Также можно посмотреть на данные через призму домена или команды и узнать, какие системы ими разрабатываются и какие компоненты ими поставляются.

Middleware Service Hub получает данные из разных источников. Например, системы мы берём из SimpleOne, а команды — из портала Insider. Также собираем данные из файла meta-info.yaml, который добавляется в корень проектов в GitLab. Чем больше заполненных файлов meta-info.yaml в ваших проектах Gitlab, тем больше мы знаем о том, какие компоненты и системы вы разрабатываете.

Из чего состоит модель



Домен

Структурное подразделение внутри компании

Система

Набор ПО для бизнес или технологической задачи

Источники

GitLab, Insider, SimpleOne, API Portal и другие источники

Команда

Группа специалистов, которые работают над созданием, развитием и улучшением продукта

Компонент

Микросервис, библиотека, модуль или другой элемент

Ресурсы

Конфигурационная единица, представляющая собой инфраструктурный элемент

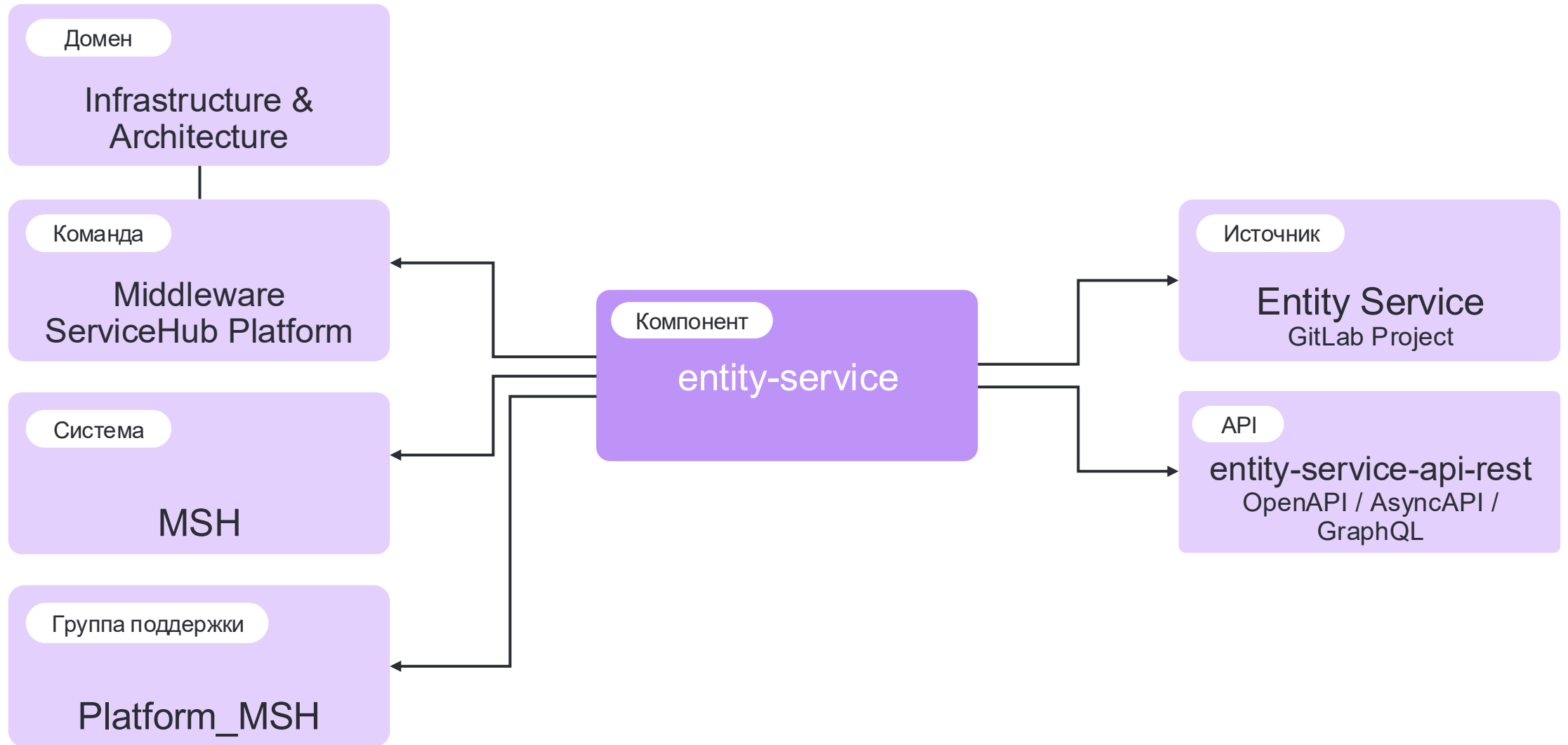
Группа поддержки

Группа поддержки системы в ITSM процессах

API

Интерфейсы, которые поставляют и потребляют компоненты

Сущности и связи между ними

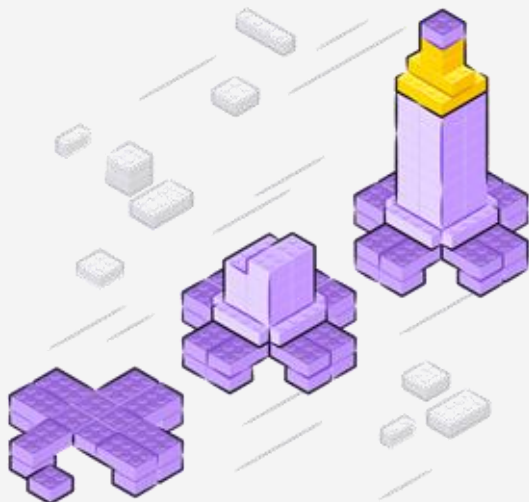


С чего всё начиналось



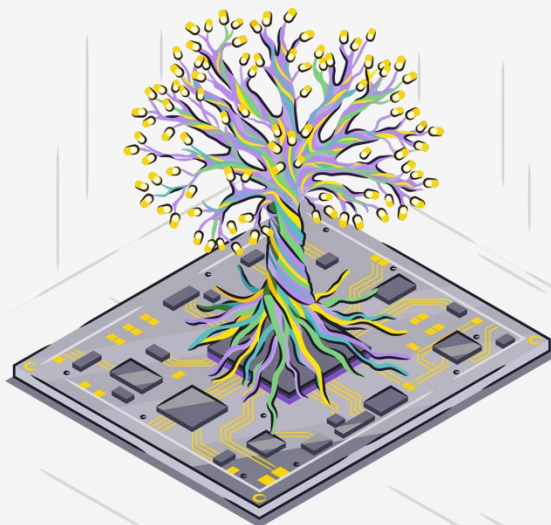
1

В 2024 году мы начали описывать компоненты в `meta-info.yaml`



2

Файл в корне проекта GitLab стал способом описывать компоненты и API как код



3

Это позволило показывать «кубики» системы, связи и владельцев рядом с инженерным процессом



meta-info.yaml — описание компонента и связей



```
api_version: 3.0.0
team_nid: "a888"
ail_id: 1234
component:
  - name: entity-service
    description: "Сервис каталога компонентов ИС"
    type: service
    lifecycle: production
    tags: [python, infra, msh]
    provides_apis:
      - name: entity-service-api-rest
        title: "API сервиса каталога MSH"
        type: "openapi"
        definitions:
          - title: test
            url: https://entity-service-test.raiffeisen.ru/openapi.json
        tags: [python, msh, infra]
```

Процесс сбора и связи данных



Собирает

Из источников
с разным
качеством данных



Обработывает

Приводит
к единой модели



Валидирует

Проверяет связи,
находит пропуски



Связывает

Строит связи
между сущностями



Визуализирует

Единая карта
ландшафта



Ценность MSH не в карточках сущностей,
а в сети связей между ними

MSH — фундамент для Digital Twin
организации: технический, архитектурный
и операционный контекст в единой модели

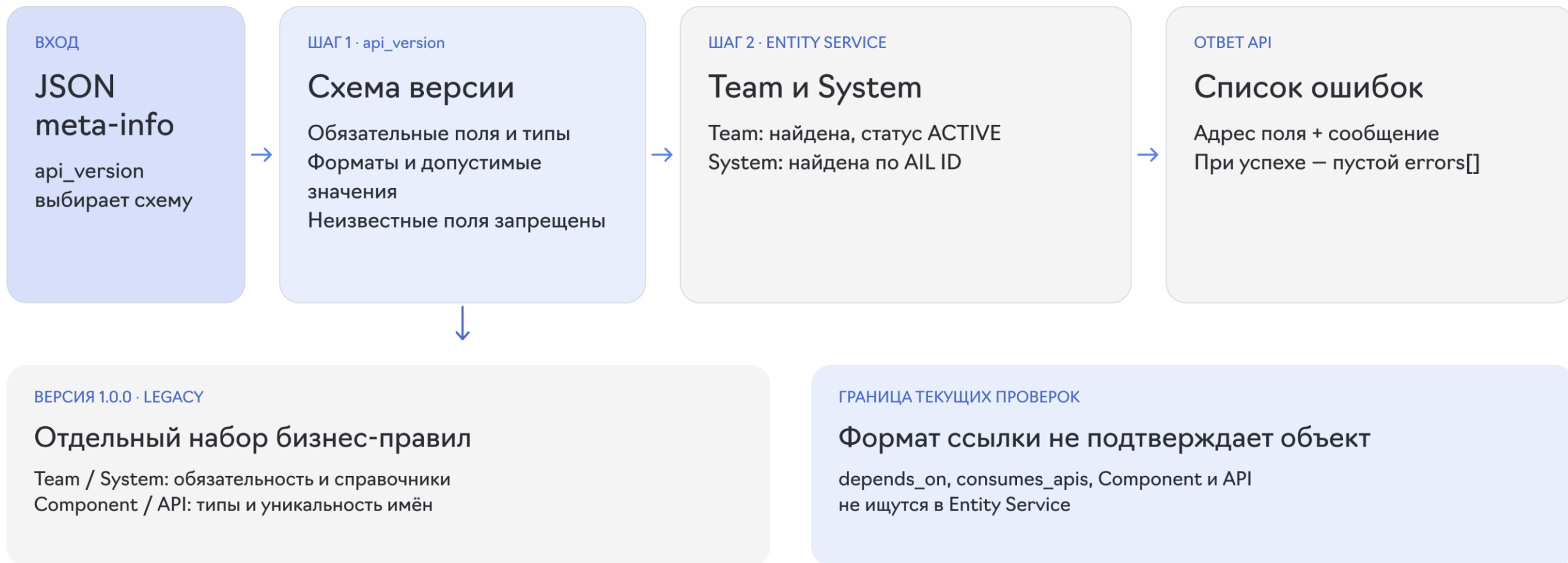


Meta-info проходит проверку схемы и справочников



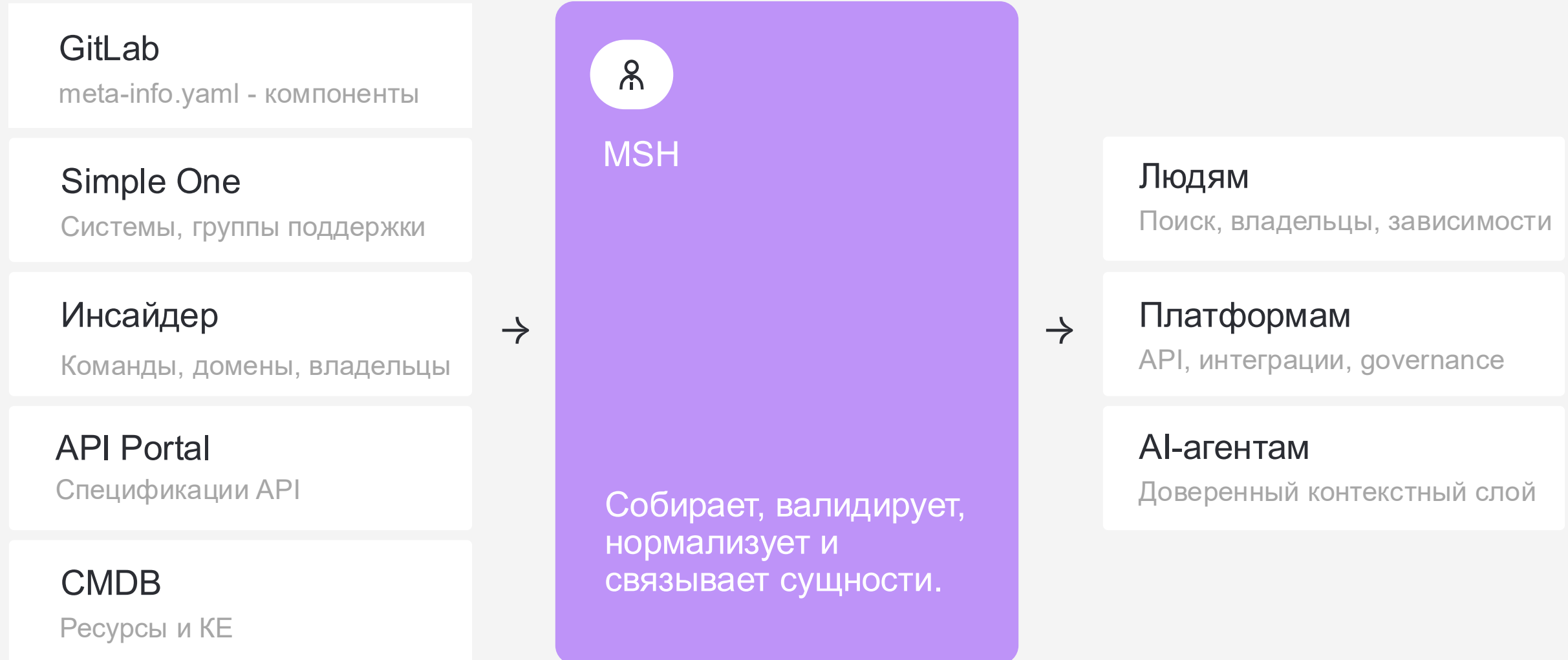
Версия спецификации определяет правила

ОСНОВНОЙ ПОТОК



Сервис принимает JSON; YAML-синтаксис проверяется на внешнем этапе.

Источники и потребители данных



Каталог компонентов



MSH

☰

DevOps Community

▼

Главная

Поиск

Аналитика

Системы

Компоненты

API

Источники

Ресурсы

Команды

Группы поддержки

Домены

Технологии и инновации > devPlatform > MSH Portal > Компоненты

🔍

Что угодно

⌘+K

На данной странице представлены компоненты информационных систем.

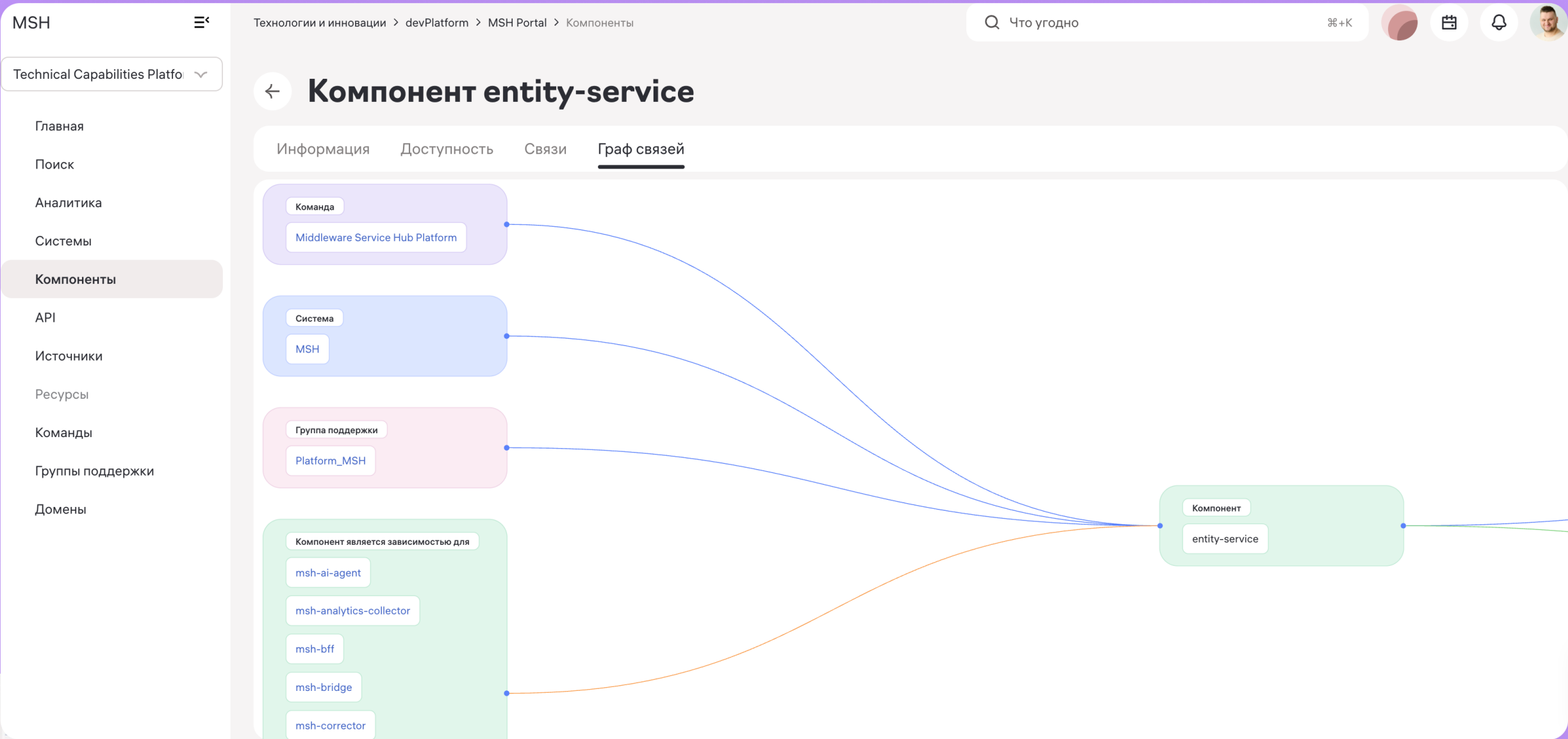
🔍

Сущность

Имя	Система	Команда владлец	Тип	Источник	Теги
1	CSPD	Core Data	Service	1	cspd falcon
1c-router	1C Connect	CDC Integrations	Service	1c-router	-
2	CSPD	Core Data	Service	2	cspd falcon
2GIS	2GIS	External Software Solutions	Service	2GIS	-
a2a-currency	FX PI API Gateway	Save Up	Service	a2a-currency	fxpi a2a ccy kotlin
a3-invoice-exporter	PUH	Everyday transfers	Service	a3-invoice-exporter	payments java
a3-notification-importer	PUH	Everyday transfers	Service	a3-notification-importer	bills java
ab-toggles	Calutron	Retail Core AI	Service	ab-toggles	python k8s
acc-accrual-schemes	Deposit Online	Liquidity Solutions	Service	Accruals	java
access-control-service	RAY	CDC Integrations	Service	access-control-service	-
access roles	devPlatform	dP Core	Service	Access Roles	typescript react Infra devPlatform dpCore



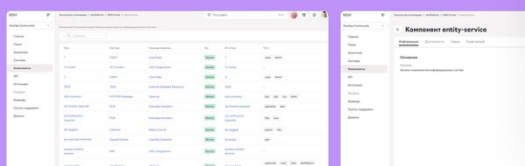
Граф связей: наглядная сеть контекста



Граф связей: наглядная сеть контекста



```
... "component_name": "entity-service",
... "related_entities": {
...   "quantities": { ...
...   },
...   "entities": [
...     {
...       "name": {
...         "title": "entity-service-api-rest",
...         "url": "https://${portal}.raiffeisen.ru/devplatform/msh/api/fafc905f-c320-43e8-8391-3deb6f332391"
...       },
...       "kind": "api",
...       "api_relationship_type": "provides_api",
...       "api_type": "openapi",
...       "api_owner": {
...         "title": "Middleware Service Hub Platform",
...         "url": "https://${portal}.raiffeisen.ru/devplatform/msh/teams/1b0246fc-d030-4a60-8568-40365dbdb0c4"
...       }
...     },
...     {
...       "name": {
...         "title": "msh-embeddings-service",
...         "url": "https://${portal}.raiffeisen.ru/devplatform/msh/components/095d40bf-9d55-4845-9f95-c99140fa597f"
...       },
...       "kind": "component",
...       "component_relationship_type": "dependency_of",
...       "component_type": "service",
...       "component_owner": {
...         "title": "Middleware Service Hub Platform",
...         "url": "https://${portal}.raiffeisen.ru/devplatform/msh/teams/1b0246fc-d030-4a60-8568-40365dbdb0c4"
...       },
...       "component_location": {
...         "title": "MSH Embeddings Service",
...         "url": "https://${portal}.raiffeisen.ru/devplatform/msh/origin/cec1a9a2-bdf5-4b1b-b569-397ce5a5d8c2"
...       }
...     },
...     {
...       "name": {
...         "title": "msh-ai-agent",
...         "url": "https://${portal}.raiffeisen.ru/devplatform/msh/components/3da1768f-7e9a-453d-8d95-c5e7c54ccb41"
...       },
...       "kind": "component",
...       "component_relationship_type": "dependency_of",
...       "component_type": "service",
...       "component_owner": {
...         "title": "Middleware Service Hub Platform",
...         "url": "https://${portal}.raiffeisen.ru/devplatform/msh/teams/1b0246fc-d030-4a60-8568-40365dbdb0c4"
...       }
...     }
...   ]
... }
```



API как контракт, а не как переписка



MSH

☰

DevOps Community

▼

Главная

Поиск

Аналитика

Системы

Компоненты

API

Источники

Ресурсы

Команды

Группы поддержки

Домены

Технологии и инновации > devPlatform > MSH Portal > API

🔍 Что угодно

⌂+K

📅

🔔

API

На данной странице представлены программные интерфейсы, которые поставляются или потребляются компонентами. Данные берутся из API Portal

🔍 entity

☑ Все API

Имя	Тип	Теги
csp-cig-block-credit-legal-entity	OpenAPI	
csp-close-legal-entity-accounts	OpenAPI	
csp-close-legal-entity-accounts-ssc	OpenAPI	
DGP Entity History Management API	OpenAPI	
entity-service-api-rest	OpenAPI	
Identity Controller	OpenAPI	
Legal Entity Card	OpenAPI	

API как контракт, а не как переписка



MSH

☰

DevOps Community

Главная

Поиск

Аналитика

Системы

Компоненты

API

Источники

Ресурсы

Команды

Группы поддержки

Домены

Технологии и инновации > devPlatform > MSH Portal > API

🔍 Что угодно

⌘+K

📅

🔔

←

API entity-service-api-rest

Информация

Спецификация

Связи

Граф связей

preview

prod

test

📄 Скачать спецификацию

Entity Service

0.1.1

OAS 3.1

service for entities handling

A service that performs the key function of the platform - processing, storing and displaying entities (business processes, systems, services, APIs, ...) as well as their metadata and specifications

[Middleware Service Hub Platform - Website](#)[Send email to Middleware Service Hub Platform](#)

Servers

https://entity-service-test.raiffeisen.ru

Authorize

v1 entities

⌵

GET

/api/v1/entities

Get Entities

🔒

✓

POST

/api/v1/entities

Post Entities

🔒

✓

MSH

☰

DevOps Community

Главная

Поиск

Аналитика

Системы

Компоненты

API

Источники

Ресурсы

Команды

Группы поддержки

Домены

API entity-service-api-rest

🔍 Что угодно

⌘+K

📅

🔔

←

API entity-service-api-rest

Информация

Спецификация

Связи

Граф связей

preview

prod

test

📄 Скачать спецификацию

Entity Service

0.1.1

OAS 3.1

service for entities handling

A service that performs the key function of the platform - processing, storing and displaying entities (business processes, systems, services, APIs, ...) as well as their metadata and specifications

[Middleware Service Hub Platform - Website](#)[Send email to Middleware Service Hub Platform](#)

Servers

https://entity-service-test.raiffeisen.ru

Authorize

v1 entities

⌵

GET

/api/v1/entities

Get Entities

🔒

✓

POST

/api/v1/entities

Post Entities

🔒

✓

MSH

☰

Technical Capabilities Platform

Главная

Поиск

Аналитика

Системы

Компоненты

API

Источники

Ресурсы

Команды

Группы поддержки

Домены

Технологии и инновации > devPlatform > MSH Portal > API

Q Что угодно

⌘+K

←

API entity-service-api-rest

Информация

Спецификация

Связи

Граф связей

Компоненты, предоставляющие API

entity-service

Компоненты, потребляющие API

msh-ai-agent

msh-bff

msh-bridge

msh-corrector

msh-embeddings-service

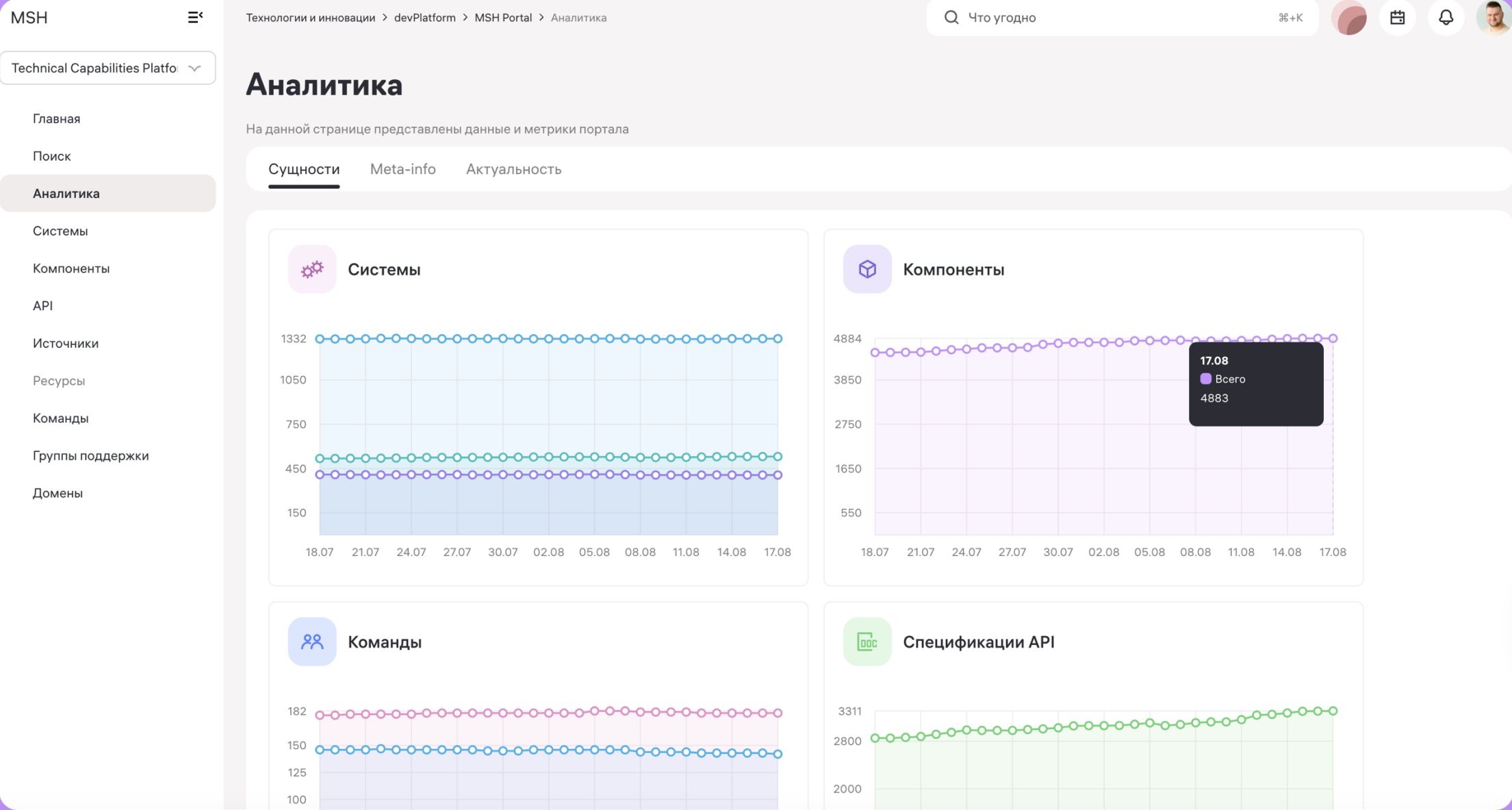
msh-mcp

msh-validator

API

entity-service-api-rest

Масштаб ландшафта сегодня



Масштаб ландшафта сегодня



4 528

Спецификаций API



1 329

Активных систем



5 619

meta-info.yaml
файлов
для описания
компонентов



194

Команды с ресурсами
из 283



5 834

Компонентов





DevOps

DGTL

В чем ценность решения?



Кому и зачем нужен MSH



Архитекторы

Карта систем, API, зависимостей и владельцев для анализа влияния изменений

Инженеры и техлиды

Быстро находят API, документацию, команды-владельцы и связанные компоненты

Руководители и платформы

Качество описаний, зрелость ландшафта, проблемные зоны, управляемость

Информационная безопасность

Прозрачность по внешним API, ресурсам, владельцам и зонам риска

AI-агенты

Структурированный источник корпоративного контекста для поиска, анализа, рекомендаций и автоматизации действий



Ключевые пользовательские сценарии



Найти API для интеграции:
где спецификация, кто
владелец, какие зависимости

Найти владельца системы,
компонента или API

Анализ влияния:
какие системы и команды
затронет изменение

Валидация meta-info.yaml
и спецификаций до релиза

Поиск неполных, устаревших
и проблемных описаний

Технологии и инновации > devPlatform > MSH Portal > Поиск

Q Что угодно

⌘+K

Поиск

Здесь можно найти информацию по компонентам и другим сущностям

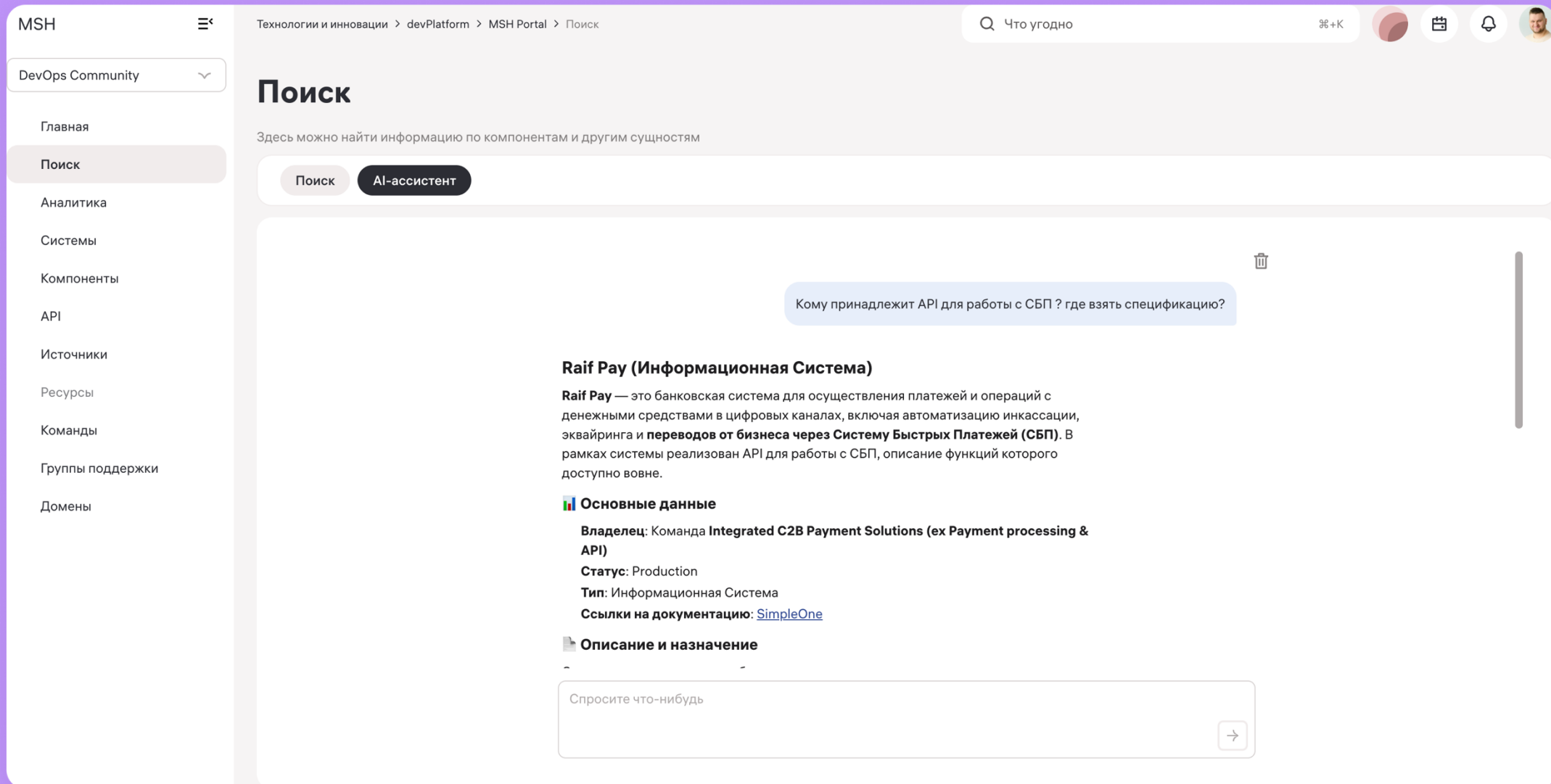
Поиск

AI-ассистент

Q сбп

Сущность

Имя	Сущность	Тип
⌵ API для отправки ИПО по C2C/ME2ME СБП	API	AsyncAPI
Спецификация API prod: отправки ИПО по C2C/ME2ME СБП , "version": "1.0.0", "description": "Асинхронное		
⌵ API интеграции B2B оплат через СБП	API	OpenAPI
⌵ API интеграции Системы быстрых платежей (СБП)	API	OpenAPI
⌵ API интеграции Системы быстрых платежей (СБП)	API	OpenAPI
⌵ API интеграции Системы быстрых платежей (СБП) для B2B	API	OpenAPI
⌵ API интеграции Системы быстрых платежей (СБП) для B2B	API	OpenAPI
⌵ Продуктовый сервис домена Диспутов СБП	API	OpenAPI
⌵ Система быстрых платежей (СБП) для B2B	API	OpenAPI
⌵ [puh-c2b-link-token-2]. API Сервиса привязки счетов для платежа без подтверждения	API	OpenAPI



Сценарий: вопрос на естественном языке — ответ на проверенном контексте MSH



MSH

☰

Технологии и инновации > devPlatform > MSH Portal > Поиск

Q Что угодно

⌘+K

DevOps Community

▼

Главная

Поиск

Аналитика

Системы

Компоненты

API

Источники

Ресурсы

Команды

Группы поддержки

Домены

Поиск

Здесь можно найти информацию по компонентам и другим сущностям

Поиск

AI-ассистент

Система также не предоставляет информацию по ресурсам, существующим для систем, так как прямо не указаны ссылки на OpenAPI спецификации (Swagger) или адреса для их получения. Описание функций API предоставляется вовне, но конкретный URL для скачивания спецификации в атрибутах отсутствует. Рекомендуется обратиться к документации в системе SimpleOne по ссылке выше или связаться с командой-владельцем.

🗑

🔗 **Связи и зависимости**

Система состоит из множества компонентов, среди которых для работы с СБП и связанными процессами выделяются:

Компоненты, связанные с СБП: `rbp-sme-sbp-qr`, `rbp-sme-sbp-connect`, `rbp-sme-sbp-cashier`, `rbp-sme-sbp-refund`, `sbp-client-registration-service`, `sbp-client-refill-service`, `sbp-docs-dms-sender`, `sbp-registration-data-collect-service`, `sbp-registration-mcp-server`.

Компоненты платежей и эквайринга: `acquiring-rfuture`, `b2b-outgoing-payment`, `b2b-invoice-service`, `b2b-payment-link`, `mf-outgoing-payments`.

Для получения точной спецификации API СБП рекомендуется запросить доступ у команды **Integrated C2B Payment Solutions**.

📄

Спросите что-нибудь

➔

Сценарий: получение счетов клиентов



Поиск

Здесь можно найти информацию по компонентам и другим сущностям

Поиск

AI-ассистент

Часть системы: Organizations.



9. account (Компонент ИС, проект Salary Product)

Назначение: Сервис счетов (в контексте зарплатных продуктов).

Владелец: Команда `Payroll&Payouts`.

Статус: Разработка (`development`).

Часть системы: Salary Product.

🚩 Итоговая рекомендация

Выбор конкретной системы зависит от типа запрашиваемых данных:

Для **запросов из ФНС** используется `zsn`.

Для получения **базовой информации из мастер-систем** — `account` (StAP) или `account` (ARS).

Для **справок** — `account-certificates`.

Для **выписок** — `account-statements`.

Для **реквизитов счетов** (корпоративный сегмент) — сервисы семейства `rbp-*`.

Для счетов по **CNUM** — `csp-cig-customer-accounts-info-cmps`.

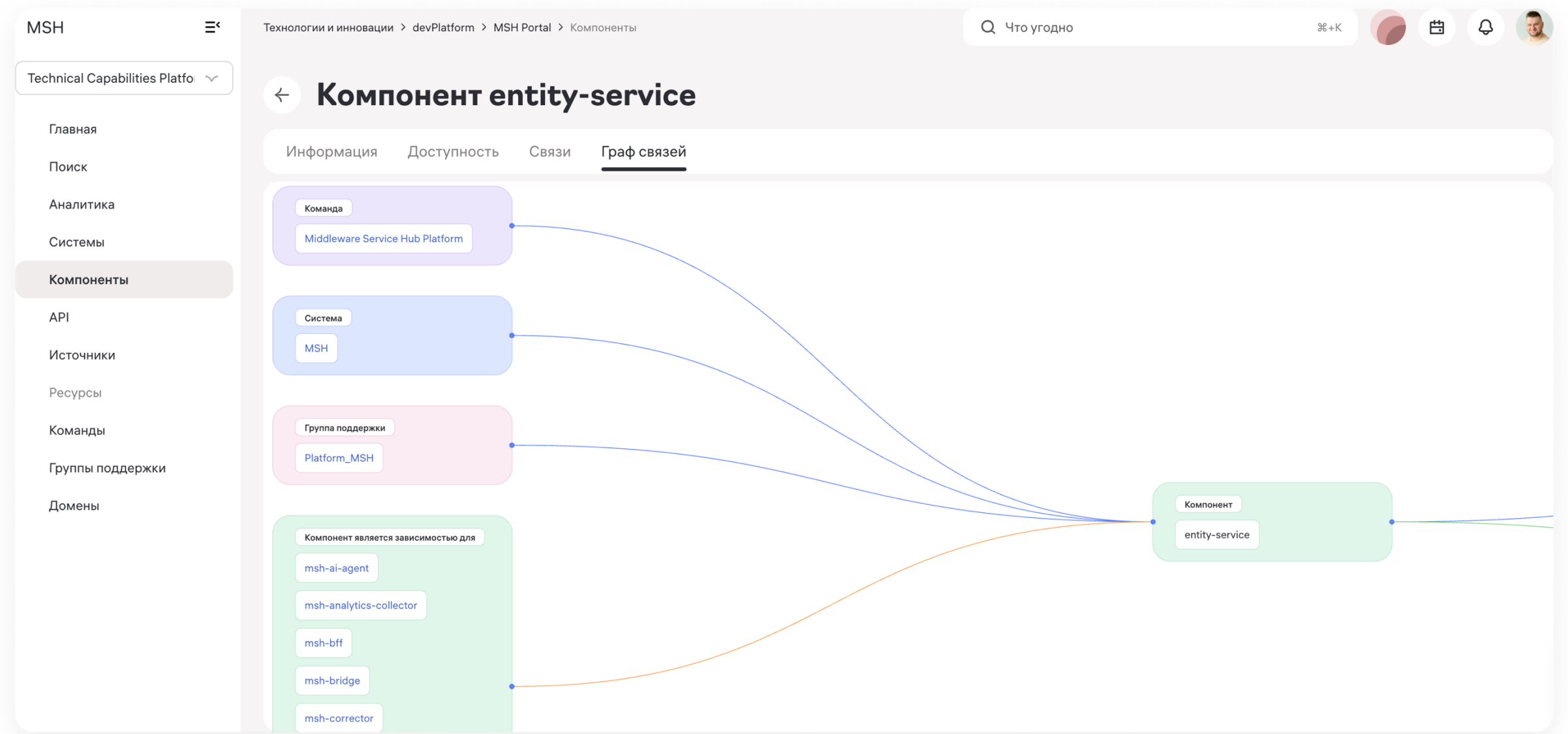
Все перечисленные компоненты находятся в статусе `production` (за исключением одного сервиса в разработке), что подтверждает их готовность к использованию.



Спросите что-нибудь



Взаимосвязи систем и интеграции



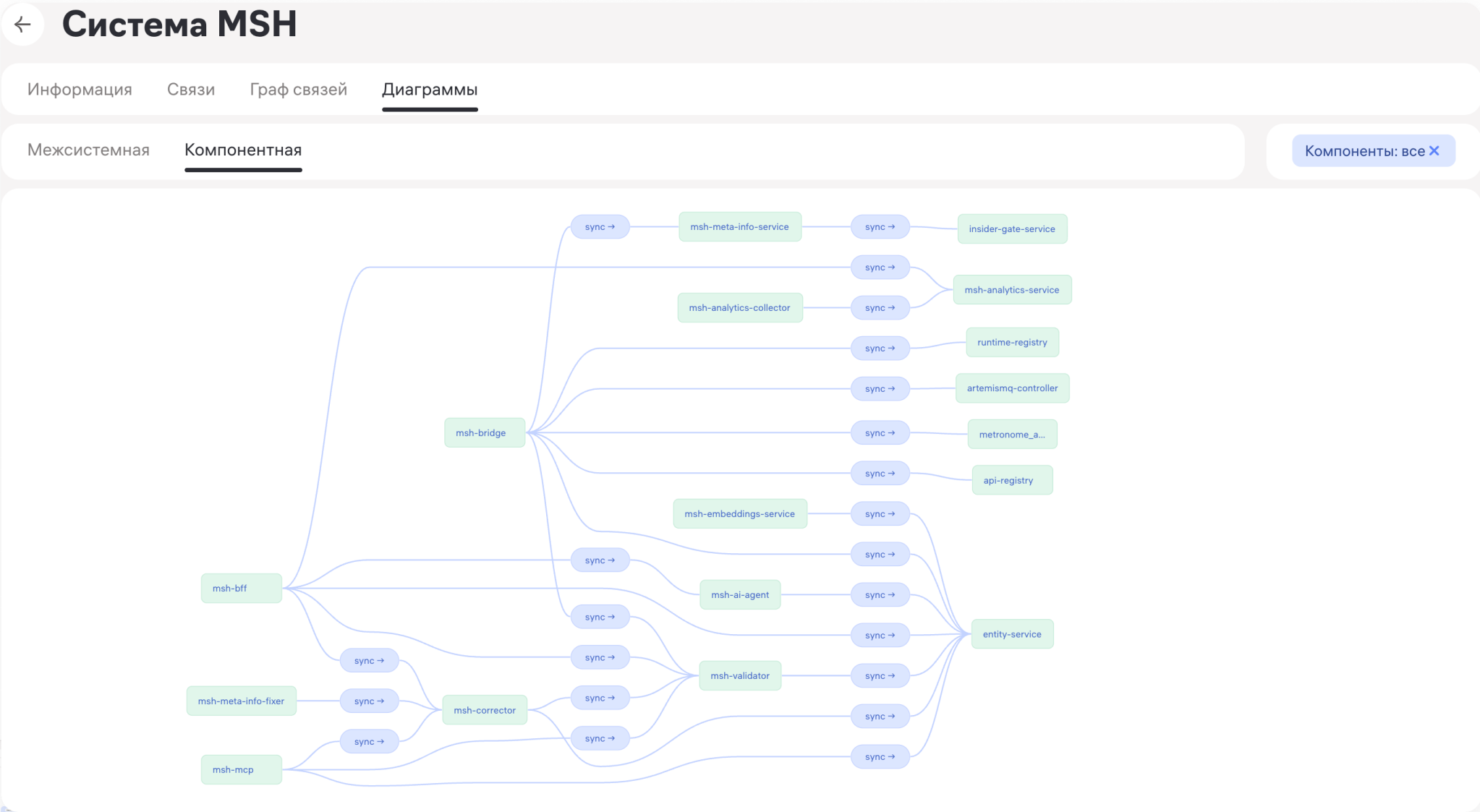
Взаимосвязи систем и интеграции



Взаимосвязи систем и интеграции



Взаимосвязи систем и интеграции



Взаимосвязи систем и интеграции



Взаимосвязи систем и интеграции



sync →

← Интеграция msh-meta-info-service → insider-gate-service

Информация

Основное

Системы участники

[TechCaps](#) - поставщик данных

[MSH](#) - потребитель данных

Компоненты интеграции

[insider-gate-service](#) - поставщик данных

[msh-meta-info-service](#) - потребитель данных

Спецификация поставщика

[Insider gate Service](#)

Дополнительно

Тип

Синхронная

Статус интеграции

неактивная

Дата синхронизации

10 октября 2026 10:30



MCP для клиентов – чтение данных



Технологии и инновации > devPlatform > MSH Portal

Что угодно

+K

Подключение MCP

Настройте подключение MCP для вашего AI-агента.

Параметры подключения

Базовый URL

https://msh-mcp.raiffeisen.ru/mcp

Команда

X-Team-Nid:

Готовая конфигурация

MCP-клиент

VS Code Copilot

Добавьте в проекте в файл .vscode/mcp.json

```
{
  "servers": {
    "msh": {
      "type": "http",
      "url": "https://msh-mcp.raiffeisen.ru/mcp",
      "headers": {

    }
  }
}
```

Как используется этот MCP?



Спроектируй интеграцию с компонентом meta Info Service ?

Сначала найду зарегистрированный `meta Info Service` в MSH и посмотрю его карточку/API-контракт. Затем предложу интеграцию, привязанную к реальным данным компонента и текущему проекту.

```
Ran search_component - msh (MCP Server)
```

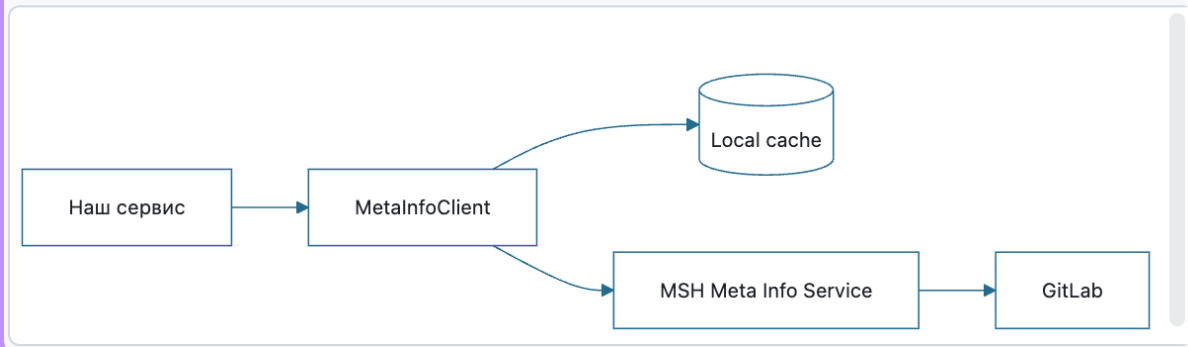
```
Ran search_component - msh (MCP Server)
```

Предлагаю интеграцию с `1438:msh-meta-info-service` как отдельный typed-adapter внутри нашего сервиса.

Цель

Получать актуальные проекты команды `F4812`, их статусы, GitLab-пути и связанные идентификаторы, не обращаясь к MSH из UI или бизнес-логики напрямую.

Схема



Как используется этот MCP?



Основной контракт

- GET /api/teams/{team_nid}/projects
 - team_nid = F4812
 - only_active=true
 - пагинация через limit и offset
- GET /api/teams/projects/{project_id}
 - получение одного проекта
- GET /api/teams/projects/summary
 - агрегированная статистика
- GET /api/teams/{team_nid}/inventory-ids
- GET /api/teams/ail-ids
- GET /api/domains

Авторизация API выполняется заголовком:

X-API-KEY: <secret>

Спроектируй интеграцию с компонентом meta Info Service ?

Сначала найду зарегистрированный meta Info Service в MSH и посмотрю его карточку/API-контракт. Затем предложу интеграцию, привязанную к реальным данным компонента и текущему проекту.

Ran_search_component - msh (MCP Server)

Ran_search_component - msh (MCP Server)

Предлагаю интеграцию с `1438:msh-meta-info-service` как отдельный typed-adaptor внутри нашего сервиса.

Цель

Получать актуальные проекты команды `F4812`, их статусы, GitLab-пути и связанные идентификаторы, не обращаясь к MSH из UI или бизнес-логики напрямую.

Схема



Как используется этот MCP?



Сформируй контракт в README.md как я могу получать данные из этого сервиса

Спроектируй интеграцию с компонентом meta Info Service ?

Сначала найду зарегистрированный `meta Info Service` в MSH и посмотрю его карточку/API-контракт. Затем предложу интеграцию, привязанную к реальным данным компонента и текущему проекту.

Ran `search_component` - msh (MCP Server)

Ran `search_component` - msh (MCP Server)

Предлагаю интеграцию с `1438:msh-meta-info-service`: как отдельный typed-adaptor внутри нашего сервиса.

Цель
Получать актуальные проекты команды `FA812`, их статусы, OILab-пути и связанные идентификаторы, не обращаясь к MSH из UI или бизнес-логики напрямую.

Схема



Основной контракт

- GET /api/teams/{team_id}/projects
 - + team_id = FA812
 - + api_key=token
 - + pagination: page: 1 или 2 и offset
- GET /api/teams/projects/{project_id}
 - + получение одного проекта
- GET /api/teams/projects/summary
 - + агрегированная статистика
- GET /api/teams/{team_id}/inventory-ids
- GET /api/teams/{id}/db
- GET /api/domains

Авторизация API выполняется заголовком:

X-API-KEY: <secret>

Как используется этот MCP?



mcp.json

README.md

Preview README.md

X

Integration with MSH Meta Info Service

Document describes obtaining data from component `1438:msh-meta-info-service`. Service collects meta-information about components from GitLab and provides data about teams and projects.

Подключение

Переменные окружения

```
export MSH_META_INFO_BASE_URL="https://<meta-info-service-host>"
export MSH_API_KEY="<api-key>"
export MSH_TEAM_NID="F4812"
```

Точный `MSH_META_INFO_BASE_URL` и значение `MSH_API_KEY` выдаёт команда-владелец сервиса (`Platform_MSH`, team NID `a794`). API-ключ нельзя хранить в Git или передавать через frontend.

Все запросы выполняются с заголовком:

```
X-API-KEY: <api-key>
```

Получить проекты команды

```
curl --fail-with-body --silent --show-error \
-H "X-API-KEY: ${MSH_API_KEY}" \
"${MSH_META_INFO_BASE_URL}/api/teams/${MSH_TEAM_NID}/projects?only_active=true&limit=1000&offset=0"
```

Параметры:

Параметр	Расположение	Тип	По умолчанию	Описание
<code>team_nid</code>	path	string	обязательный	NID команды, например <code>F4812</code>
<code>only_active</code>	query	boolean	<code>true</code>	Вернуть только активные проекты
<code>limit</code>	query	integer/null	<code>1000</code>	Максимальное число проектов
<code>offset</code>	query	integer/null	<code>0</code>	Смещение для пагинации

Спроектируй интеграцию с компонентом `1438:msh-meta-info-service`

Сначала найду зарегистрированный `meta Info Service` в MS. Затем предложу интеграцию, привязав её к компоненту и текущему проекту.

Ran `search_component` - msh (MCP Server)

Ran `search_component` - msh (MCP Server)

meta Info Service

GitLab

X-API-KEY: <secret>

1438:msh-meta-info-service

Platform_MSH

team NID a794

https://meta-info-service-host

api/teams/F4812/projects?only_active=true&limit=1000&offset=0

Как используется этот MCP?



Пример ответа:

```
[
  {
    "project_id": 12345,
    "full_path": "group/project",
    "name": "project",
    "is_active": true,
    "team_nid": "F4812",
    "updated_at": "2026-09-30T12:00:00Z"
  }
]
```

Ответ может содержать дополнительные поля, включая идентификаторы связанных систем. Клиент должен игнорировать неизвестные поля.

Получить один проект

```
curl --fail-with-body --silent --show-error \
  -H "X-API-KEY: ${MSH_API_KEY}" \
  "${MSH_META_INFO_BASE_URL}/api/teams/projects/${PROJECT_ID}"
```

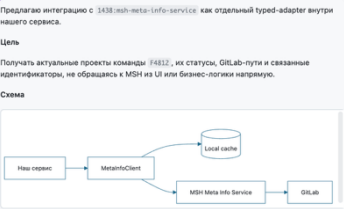
`PROJECT_ID` — числовой идентификатор из списка проектов.

Спроектируй интеграцию с компонентом meta Info Service ?

Сначала найду зарегистрированный meta Info Service в MSH и посмотрю его карточку/API-контракт. Затем предложу интеграцию, привязанную к реальным данным компонента и текущему проекту.

Ran_search_component = msh (MCP Server)

Ran_search_component = msh (MCP Server)



Основной контракт

- GET /api/teams/{team_id}/projects
 - + team_id = F4812
 - + api_key=token
 - + pagination: offset (100) и offset
- GET /api/teams/projects/{project_id}
- GET /api/teams/projects/summary
 - + агрегированная статистика
- GET /api/teams/{team_id}/inventory-ids
- GET /api/teams/{id}
- GET /api/domains

Авторизация API выполняется заголовком:

X-API-KEY: <secret>

Сформируй контракт в README.md как я могу получать данные из этого сервиса



Context Annotation Platform



Context Annotation Platform

КНARCHENKO Evgeny

Выйти

Редактировать

Поиск бизнес-процесса

Команды

Technical Capabilities Platform

Управление командным доступом

Бизнес-процессы

Реестр метаинформации

Шаги бизнес-процесса

Запуск синхронизации из GitLab

Запуск синхронизации для Home Search

Получение списка доменов

Получение списка команд

Получение сводки по командам

Получение метаданных проектов

Получение сводки по проектам

Сохранение/обновление проекта

Сервисы

Data-контракты

API

Unbridled RnD Group

devPlatform

Бизнес-процесс «Реестр метаинформации»

AI readiness not checked

Заголовок:

Реестр метаинформации

Код бизнес-процесса:

project_meta_registry

Tier:

Tier 2 — Important

Описание:

Сбор и агрегация метаданных о проектах из GitLab, Jira, SonarQube и связанных систем.

Insider Wiki:

Команды, которые отвечают за процесс:

Technical Capabilities Platform

Сервисы, в которых реализуется процесс:

Meta Info Service

Тест-кейсы процесса:

Покрытие шагов:

0/8

Без покрытия: meta-info-sync-gitlab-sync-trigger, meta-info-sync-home-search-sync-trigger, catalog-domains-

Канал Context Annotation Platform в MM

Оставить обратную связь

Context Annotation Platform



Context Annotation Platform

КНARCHENKO Evgeny

Выйти

График

Информация

Команды

Technical Capabilities Platform

Управление командным доступом

Бизнес-процессы

Реестр метаданных

Шаги бизнес-процесса

Запуск синхронизации из GitLab

Запуск синхронизации для Home Search

Получение списка доменов

Получение списка команд

Получение сводки по командам

Получение метаданных проектов

Получение сводки по проектам

Сохранение/обновление проекта

Сервисы

Data-контракты

API

Управление командным доступом

Шаг «Получение списка команд»

Заголовок:Получение списка команд

Код шага:catalog-teams-read

Insider link:— не указано —

Описание:Возвращает полный список команд с проектами

Ручное выполнение

Требуется создание SLI?Да Нет

Нужна авторизация?Да Нет

Каналы, через которые реализуется процесс:

Сервисы:Meta Info Service

Тест-кейсы:

Показатели доступности (SLI):

Процессы, в которых используется шаг:Реестр метаданных

Редактировать

Поиск бизнес-процесса

Канал Context Annotation Platform в MM

Оставить обратную связь

Как спецификация «прорастает» в инженерные процессы



```
api_version: 3.0.0
team_nid: "a888"
ail_id: 1234
component:
  - name: entity-service
    description: "Сервис каталога компонентов ИС"
    type: service
    lifecycle: production
    tags: [python, infra, msh]
    provides_apis:
      - name: entity-service-api-rest
        title: "API сервиса каталога MSH"
        type: "openapi"
        definitions:
          - title: test
            url: https://entity-service-test.raiffeisen.ru/openapi.json
        tags: [python, msh, infra]
```

Как спецификация «прорастает» в инженерные процессы



```
define-meta-info:
  stage: prepare
  image: $CLI_TOOLS_IMAGE
  tags: [$SMALL_JOB]
  script:
    - echo "AIL_ID=$(yq '.ail_id // 0' meta-info.yaml)" >> META_INFO.env
    - echo "COMPONENT_NAME=$(yq '.component[0].name' meta-info.yaml)" >> META_INFO.env
    - echo "GITLAB_PROJECT_ID=${CI_PROJECT_ID}" >> META_INFO.env
    - cat META_INFO.env
  artifacts:
    reports:
      dotenv:
        - META_INFO.env
  interruptible: true
```

```
ail: meta-info: 1.0.0
name: "ail"
description: "ail"
image: $CLI_TOOLS_IMAGE
tags: [$SMALL_JOB]
script:
  - echo "AIL_ID=$(yq '.ail_id // 0' meta-info.yaml)" >> META_INFO.env
  - echo "COMPONENT_NAME=$(yq '.component[0].name' meta-info.yaml)" >> META_INFO.env
  - echo "GITLAB_PROJECT_ID=${CI_PROJECT_ID}" >> META_INFO.env
  - cat META_INFO.env
artifacts:
  reports:
    dotenv:
      - META_INFO.env
interruptible: true
```

Как спецификация «прорастает» в инженерные процессы



```
# Standard metadata annotations applied to every Kubernetes resource.
# See: https://confluence.raiffeisen.ru/... (internal standards)
meta:
  projectId: ""          # gitlabci.raiffeisen.ru/project.id  – GitLab project ID
  componentName: ""     # meta-info.raiffeisen.ru/component.name – service/component name
```

```
{{/*
Standard metadata annotations for resource identification.
Set meta.projectId and meta.componentName in values.yaml.
*/}}
{{- define "universal-python.metaAnnotations" -}}
gitlabci.raiffeisen.ru/project.id: {{ .Values.meta.projectId | quote }}
meta-info.raiffeisen.ru/component.name: {{ .Values.meta.componentName | quote }}
{{- end }}
```

```
gitlabci.raiffeisen.ru/project.id: ""
meta-info.raiffeisen.ru/component.name: ""
```

```
define-meta-info:
  stage: prepare
  image: gitlab-ci-tools:latest
  tags: [small, job]
  script:
    - echo "All IDs (eg. 'all_id // 0' meta-info.yaml)" >> META_INFO.env
    - echo "COMPONENT_NAME (eg. 'component[0].name' meta-info.yaml)" >> META_INFO.env
    - echo "GITLAB_PROJECT_ID (eg. 'gitlab-project-id' meta-info.yaml)" >> META_INFO.env
    - cat META_INFO.env
  artifacts:
    reports:
      dotenv:
        - META_INFO.env
  interruptible: true
```

Как спецификация «прорастает» в инженерные процессы



MSH

☰

Технологии и инновации > devPlatform > MSH Portal > Компоненты

🔍 Что угодно

⌘+K

🗓

🔔²

Technical Capabilities Platform

Главная

Поиск

Аналитика

Системы

Компоненты

API

Источники

Ресурсы

Команды

Группы поддержки

Домены

←

Компонент meta-info-service

ИнформацияДоступностьСвязиГраф связейRuntime

ОкружениеТип экземпляраenv R&R

PROD

k8s Deployment

meta-info-service

Ресурс techcaps-004.kaas.raiffeisen.ru

Пространство meta-info-service

11.09.2026, 13:48:44

```
apiVersion: v1
kind: Pod
metadata:
  name: meta-info-service
  namespace: meta-info-service
spec:
  containers:
  - name: meta-info-service
    image: techcaps-004.kaas.raiffeisen.ru/meta-info-service:11.09.2026, 13:48:44
```

```
define-meta-info:
  stage: prepare
  image: techcaps-004.kaas.raiffeisen.ru/meta-info-service:11.09.2026, 13:48:44
  tags: [SMALL_IMAGE]
  script:
    - echo "All IDs (eg. 'id' or 'id' meta-info.yaml) >> META_INFO.env"
    - echo "COMPONENT_NAME (eg. 'component[is_name]' meta-info.yaml) >> META_INFO.env"
    - echo "GITLAB_PROJECT_ID (eg. 'gitlab-project-id') >> META_INFO.env"
    - cat META_INFO.env
  artifacts:
  - name: meta-info-service-logs
    format: tar
    paths:
    - META_INFO.env
  interruptible: true
```

```
# Standard metadata annotations applied to every Kubernetes resource.
# See: https://confluence.raiffeisen.ru/... (internal standards)
meta:
  projectId: "" # gitlabci.raiffeisen.ru/project.id - GitLab project ID
  componentName: "" # meta-info.raiffeisen.ru/component.name - service/component name
```

```
#!/bin/bash
Standard metadata annotations for resource identification.
Set meta.projectId and meta.componentName in values.yaml.
*/}
{{- define "universal-python.metaAnnotations" -}}
gitlabci.raiffeisen.ru/project.id: {{ .Values.meta.projectId | quote }}
meta-info.raiffeisen.ru/component.name: {{ .Values.meta.componentName | quote }}
{{- end }}
```

The screenshot shows the 'Komoot meta-Info-service' interface. The left sidebar contains a navigation menu with 'Meta-Info' selected. The main content area has tabs for 'Information', 'Settings', 'Email', and 'Settings'. The 'Information' tab is active, displaying user details for 'johndoe' (ID: 123456789). The details include the user's name, email, phone number, and a list of devices. The page also features a 'Logout' button and a 'Settings' link.

```

{/**
Standard metadata annotations for resource identification.
Set meta.projectId and meta.componentName in values.yaml.
*/}}

{{- define "universal-python.metaAnnotations" -}}
gitlabci.raiffeisen.ru/project.id: {{ .Values.meta.projectId | quote }}
meta-info.raiffeisen.ru/component.name: {{ .Values.meta.componentName | quote }}
{{- end }}

```

Как спецификация «прорастает» в инженерные процессы



Мониторинг Архитектура Метрики SLI Производительность Зависимости

Статус компонентов

Общее состояние: **96.4%**

Payment Processing System

System

Availability 99.6% — Health: 96.9%

Core Banking System

System

Availability 99.1% — Health: 97.1%

Transaction Engine

Component

Latency 85.2ms ⚠ Availability 98.9% — Latency 99.3% ✓ Health: 85.2%

Account Service

Component

Availability 100.0% — Availability 97.7% ✓ Latency 98.9% — Health: 100.0%

Core Banking API

Api

Availability 100.0% — Health: 100.0%

Customer Management Svsstem

System

Легенда статусов

Здоровое состояние (≥95%) Предупреждение (90-95%) Критическое состояние (<90%)

```
apiVersion: v1
kind: Pod
metadata:
  labels:
    app: raiffeisen
  name: raiffeisen
spec:
  containers:
    - name: raiffeisen
      image: raiffeisen/raiffeisen:1.0.0
      ports:
        - containerPort: 8080
```

```
define-meta-info:
  stage: prepare
  image: raiffeisen/raiffeisen:1.0.0
  tags: [SMAI_L100]
  script:
    - echo "All IDs (eg 'sl_id // 8' meta-info.yaml)" >> META_INFO.env
    - echo "COMPONENT_NAME (eg 'component[0].name' meta-info.yaml)" >> META_INFO.env
    - echo "SYSTEM_PROJECT_ID (eg 'project_id' meta-info.yaml)" >> META_INFO.env
    - cat META_INFO.env
  reports:
    - dotenv:
        META_INFO.env
  interruptible: true
```

```
# Standard metadata annotations applied to every Kubernetes resource.
# See: https://confluence.raiffeisen.ru/... (internal standards)
meta:
  projectId: "" # gitlabci.raiffeisen.ru/project.id - GitLab project ID
  componentName: "" # meta-info.raiffeisen.ru/component.name - service/component name
```

```
#!/usr/bin/env python
Standard metadata annotations for resource identification.
Set meta.projectId and meta.componentName in values.yaml.
*/}
{{- define "universal-python.metadataAnnotations" -}}
gitlabci.raiffeisen.ru/project.id: {{ .Values.meta.projectId | quote }}
meta-info.raiffeisen.ru/component.name: {{ .Values.meta.componentName | quote }}
{{- end }}
```

meta-info.yaml — что скрывается за id ?



```
api_version: 3.0.0
team_nid: "a888"
ail_id: 1234
component:
  - name: entity-service
    description: "Сервис каталога компонентов ИС"
    type: service
    lifecycle: production
    tags: [python, infra, msh]
    provides_apis:
      - name: entity-service-api-rest
        title: "API сервиса каталога MSH"
        type: "openapi"
        definitions:
          - title: test
            url: https://entity-service-test.raiffeisen.ru/openapi.json
        tags: [python, msh, infra]
```

meta-info.yaml — что скрывается за id ?



ail_id: 1234

AIL - Реестр ИТ-систем

Редактировать

Сохранить

Сохранить и выйти

1438: MSH

ITAM: v1.29.0

AIL ID ?	Тип системы ?	Уровень доступности (Tier) ?	Инструкция по редактированию и согласованию: Перейти Перейти на портал Сервис-деск: 1.91. Регистрация ПО. Изменение сведений в Реестре ПО Внимание! Изменение Бизнес-владельца, Зам. бизнес-владельца, ИТ-владельца, Архитектора, Команды-владельца, Группы поддержки, Эксперта, Заместителя эксперта наследуются во все четыре среды ИТ-системы.
1 438	General Application	Tier 2	
Наименование ?	Статус ?	Категория значимости объекта КИИ ?	
MSH	Production	Is not CII	
Полное наименование ?	Видение архитектуры ?	Уровень критичности ?	
Middleware Service Hub	Current	Non Critical	
Slug метка ?		Класс целостности данных ?	
msh		Не определено	
Альтернативное наименование ?		Уровень доступности (RBI) ?	
Каталог компонентов информационных систем, Каталог сервисов		Не определено	
Описание системы ?		IT Risk Assessment	
Система, собирающая, обрабатывающая и хранящая данные о компонентах информационных систем и строящая связи между ними и другими сущностями нашей организации.		Выберите значение	

Основная информация

Даты жизненного цикла

Информационная безопасность

Доступность

Описание ENG

Техническая информация

История

Про ценность - скорость поиска ресурсов



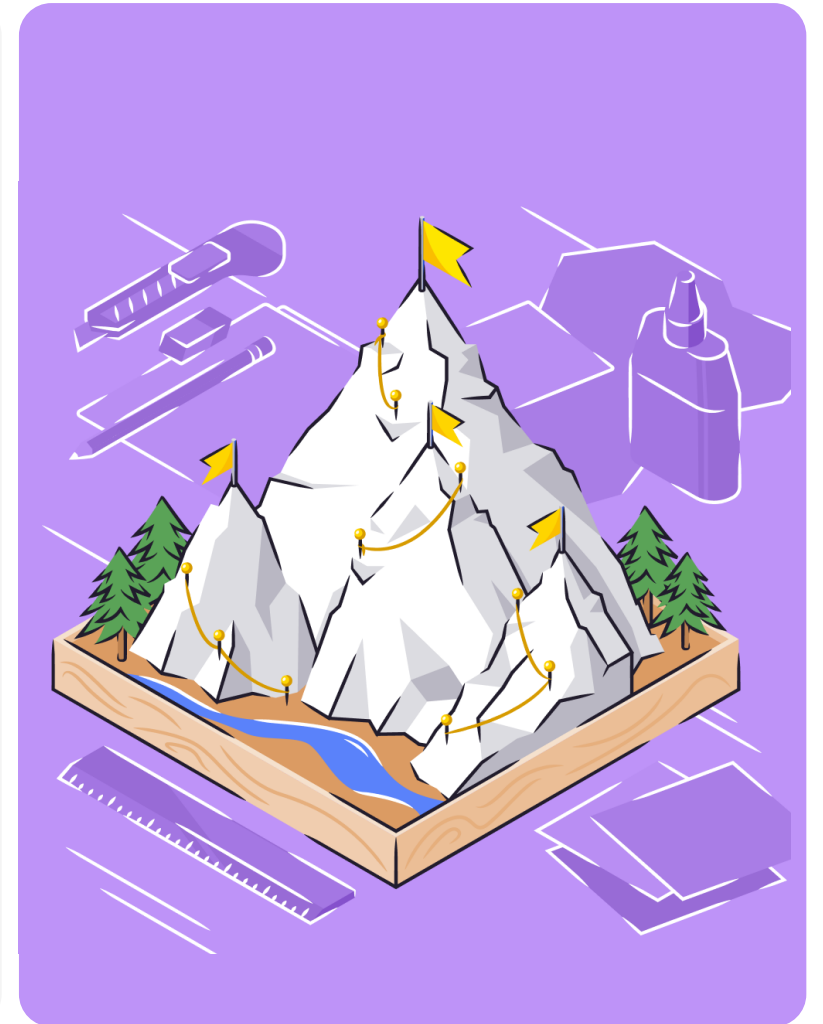
Часы / Дни

Поиск ресурсов занимал долгое время:

- Переписка в чате
- Поиск документации
- Чтение репозиторий
- Преодоление коммуникационных барьеров

Минуты

Теперь, поиск ресурсов – это просто, запрос





Какие были челленджи?



Договориться о едином формате



Разные команды описывали свою мета информацию по-своему либо вовсе не описывали



Договорились, что meta-info.yaml это единый формат описания и этим надо заниматься 😊



Сейчас покрытие meta-info.yaml составляет 69%



Низкое качество и неполнота данных



Собрать правдивую, актуальную и связанную модель из источников разного качества



Валидация данных и связей, исправление пропусков

Автообновление meta-info — ручные процессы делаем автоматическими



Текущая актуальность meta-info в компании составляет 55%



Встроить в реальные процессы



Интегрировать работу с meta-info.yaml в процесс работы команд



Мы стали продвигать заполнение файла в нативном формате для инженеров и быть ближе к коду при управлении метаданной



Построена интеграция со всеми системами инвентаризации





А что со всем этим делать?

Как и с чего начать?



Как трансформировать
все это в пользу?

Что начать делать?



Начать строить подход к разметке данных

Сейчас это особо важно в свете развития AI



Попробовать развернуть backstage для экспериментов по разметке данных



Масштабировать подход к разметке ресурсов в организации



Что важно не потерять при реализации



01

Качество данных:
источник должен быть
доверенным, а данные —
валидируемыми

02

Владение:
за сущности и описания
должны отвечать
реальные команды

03

Встраивание:
развитие и актуализации
разметки должна жить
в процессах разработки,
архитектуры,
ИБ и эксплуатации

Видение эволюции контекста с MSH



Каталог

Системы, компоненты, API и владельцы в одном месте

Карта связей

Зависимости, интеграции, источники и ownership

Digital Twin

Связь с бизнес-архитектурой, value streams, SLA/SLO и изменениями

Architecture as Code

Описание ландшафта становится частью инженерного процесса

AI Harness

Управляемая среда для агентов поверх проверенного контекста

Governance

Контроль качества, полноты и актуальности описаний

Полная автоматизация

Организация лучше понимает саму себя и автоматизирует больше решений

Контекст как инфраструктура

MSH становится базовым слоем для людей, платформ и AI

Что унести с собой



Хаос — это не отсутствие данных

Проблема в отсутствии связанного и доверенного контекста



MSH превращает данные в карту

Системы, API, команды и интеграции становятся частью живой модели организации



Это база для Digital Twin

Architecture as Code, API governance и связанный технологический ландшафт



AI Harness требует контекста

Агентам нужен управляемый доступ к проверенной корпоративной модели





БОЛЬШЕ
КОНТЕКСТА
=
БОЛЬШЕ
ВОЗМОЖНОСТЕЙ

КАК ПРЕВРАТИТЬ ХАОС В КОНТЕКСТ, РАЗВИВАЯ **HARNESS**

MSH, Digital Twin, AI Harness и Scaffolding



КОНТЕКСТ

- Инженеры были хранителями знаний
- Документация: GitLab, Confluence, runbooks
- Связи между системами жили в головах людей
- Если что-то ломалось — разбираться приходилось DevOps'am

ПРОБЛЕМА



- Знание об организации разрознено и устаревае
- Данные размазаны по GitLab, Wiki, чатам, Service Management
- Интеграции замедляются
- Архитектурные решения принимаются на неполной картине
- Владельцев ищут вручную
- AI без контекста слеп

ГДЕ
ЭТО?
КТО?
КАК СВЯЗАНО?

HARNESS И SCAFFOLDING



HARNESS
(среда)

КОНТЕКСТ
+ ИНСТРУМЕНТЫ
+ ПРАВИЛА
+ ДОСТУПЫ

- Если модель — это процессор, то harness — среда вокруг него
- Harness = управляемая рабочая среда для модели
- Scaffolding = сценарии, orchestration, шаги, логика работы агента
- Контекст нужен не только людям, но и ассистентам, агентам и автоматизированным сценариям

ПРАВИЛЬНЫЙ
КОНТЕКСТ
ДВИГАЕТ
AI ВПЕРЕД

МОДЕЛЬ

- Домен
- Команда
- Система
- Компонент
- API
- Ресурс
- Источник
- Группа поддержки

ОДНА
МОДЕЛЬ
=
МНОГО
СВЯЗЕЙ



meta-info.yaml —
компоненты и API как код

МАСШТАБ

- 4 528 компонентов
- 5 619 активных систем
- 5 834 API-спецификаций
- 1 329 meta-info.yaml файлов
- 69% покрытие meta-info
- 55% актуальность meta-info

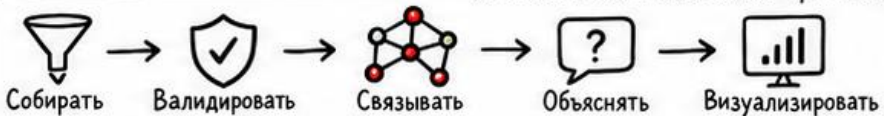
ЧЕЛЛЕНДЖИ

- Договориться о едином формате
- Низкое качество и неполнота данных
- Встроить работу с meta-info.yaml в реальные процессы

69%
покрытие
55%
актуальность

Мiddleware Service Hub (MSH)

слой контекста системной архитектуры



Ценность MSH не в карточках, а в сети связей между ними



MSH —
фундамент
для Digital Twin
организации

ИСТОЧНИКИ → MSH → ПОТРЕБИТЕЛИ



ОДНИ
ДААННЫЕ.
МНОГО
ПОЛЬЗЫ

ЦЕННОСТЬ

- Архитекторы — карта систем, API, зависимостей и владельцев
- Инженеры и техлиды — быстро находят API, документацию и команды-владельцы
- ИБ — прозрачность по внешним API, ресурсам и зонам риска
- Руководители и платформы — качество описаний и зрелость ландшафта
- AI-агенты — структурированный контекст для поиска, анализа, рекомендаций и автоматизации

Поиск ресурсов:
было часы/дни
→ стало минуты

КОНТЕКСТ
ДАЕТ
ОТВЕТЫ!

ЧТО НАЧАТЬ ДЕЛАТЬ

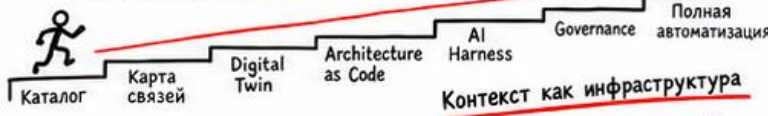
- Начать строить подход к разметке данных
- Масштабировать подход к разметке ресурсов в организации
- Попробовать Backstage для экспериментов по разметке данных

Сейчас это
особенно важно
в свете
развития AI

НЕ ПОТЕРЯТЬ

- Качество данных: источник должен быть доверенным, а данные — валидируемыми
- Владение: за сущности и описания должны отвечать реальные команды
- Встраивание: разметка должна жить в процессах разработки, архитектуры, ИБ и эксплуатации

ЭВОЛЮЦИЯ



Контекст как инфраструктура

Хаос — это не отсутствие данных. Проблема в отсутствии связанного и доверенного контекста.

БОЛЬШЕ
КОНТЕКСТА
БОЛЬШЕ
РЕЗУЛЬТАТЫ

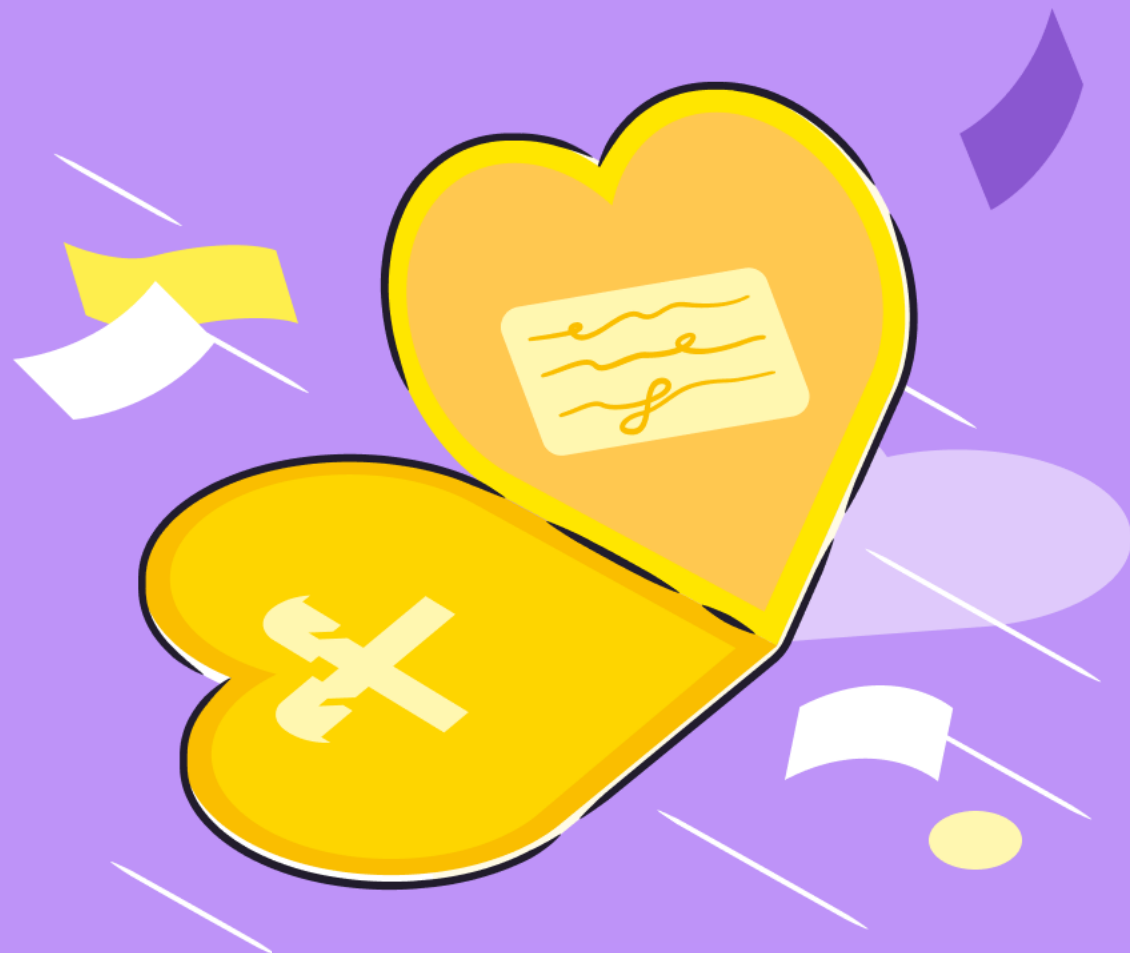
СТРОЙТЕ СВОЙ HARNESS И SCAFFOLDING!

AI будет настолько хорош, насколько хорош контекст, в котором мы позволяем ему действовать. Поэтому строить Harness и Scaffolding — значит строить среду, в которой интеллект становится полезным.



DevOps

DGTL



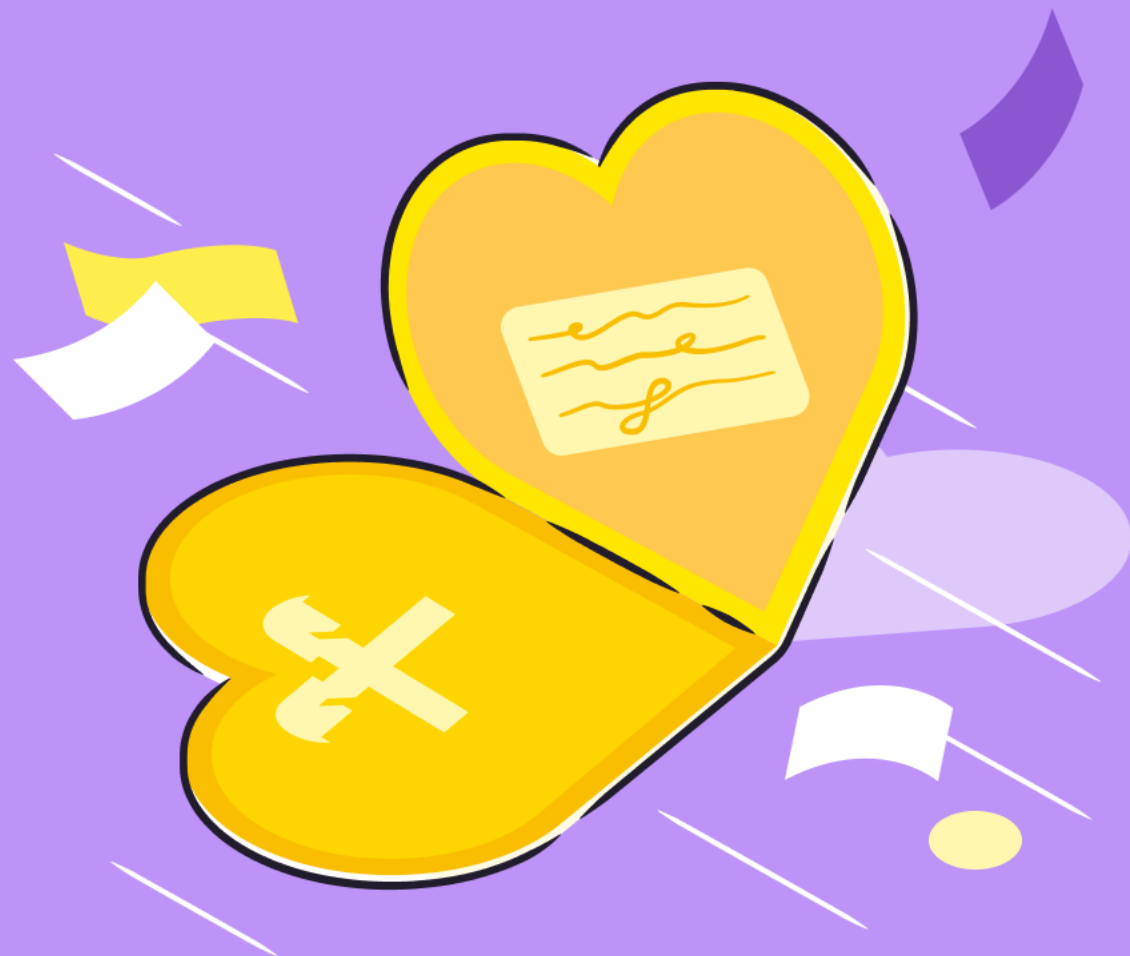
Стройте свой Harness

AI будет настолько хорош, насколько хорош контекст, в котором мы позволяем ему действовать. Поэтому строить Scaffolding — значит строить среду, в которой интеллект становится полезным



DevOps

DGTL



Стройте свой Harness и Scaffolding!

AI будет настолько хорош, насколько хорош контекст, в котором мы позволяем ему действовать. Поэтому строить Scaffolding — значит строить среду, в которой интеллект становится полезным